Please edit and update with any useful data resources, cool links, or other resources you'd like to share.

SI 601 Project Resources

I’ve gathered a set of a few Web resources and fun links that you might find fun try while exploring project ideas. Disclaimer: I’ve tried to make these as up-to-date as possible but some of these may be stale.

These resources are generally Web APIs that you can call by building and fetching a special kind of URL, which the server will process and return results in XML, JSON, or some other structured format. They provide everything from social media streams to image metadata to question-answering.

Some of the APIs require that users register and get a special “API key” string that you embed in the URL when you make an API call. In one or two cases, I’ve registered a key for the whole class to use and noted it in the description.

This is just a small (but high-quality) sample of what’s out there. Please do share any cool resources you find with your fellow students in the CTools public forum.

**Meta-resources**

external link: [http://www.programmableweb.com/apis/directory (Links to an external site.)](http://www.programmableweb.com/apis/directory)

This is the **Programmable Web** meta-directory I showed you in the first lecture that has thousands of data sources. I recommend starting here.

This website **has dozens of public datasets** - some fun, some a bit, well.. quirky. external link: [http://rs.io/2014/05/29/list-of-data-sets.html (Links to an external site.)](http://rs.io/2014/05/29/list-of-data-sets.html)

The **Academic Torrents site** has a growing number of datasets, including a few text collections that might be of interest to some of you (Wikipedia, email, twitter, academic, etc) for current or future projects.

external link: [http://academictorrents.com/browse.php?cat=6 (Links to an external site.)](http://academictorrents.com/browse.php?cat=6)

**Wolfram | Alpha API** A computational knowledge engine with question-answering abilities and access to large databases of the world’s facts

Experiment with the urllib2 module and a URL like this: external link: [http://api.wolframalpha.com/v2/query?input=XXXXX&appid=459LEL-W3KJ88H8Y7 (Links to an external site.)](http://api.wolframalpha.com/v2/query?input=XXXXX&appid=459LEL-W3KJ88H8Y7)

For example: external link: [http://api.wolframalpha.com/v2/query?input=canada%20population&appid=459LEL-W3KJ88H8Y7 (Links to an external site.)](http://api.wolframalpha.com/v2/query?input=canada%20population&appid=459LEL-W3KJ88H8Y7)

where you replace the XXXXX with what you want to know. NOTE: the ‘&appid=459LEL-W3KJ88H8Y7’ part is vital; it is a W|A AppID I got for the class. Feel free to use that one.

You can get your own and read more about the API, here: external link: [http://products.wolframalpha.com/api/ (Links to an external site.)](http://products.wolframalpha.com/api/)

And you can explore how it works here: external link: [http://products.wolframalpha.com/api/explorer.html (Links to an external site.)](http://products.wolframalpha.com/api/explorer.html)

**Visualization Resources**

Simple interactive timelines using TimelineJS. external link: [http://timeline.knightlab.com/ (Links to an external site.)](http://timeline.knightlab.com/)

Geographic charts using OpenHeatMap. external link: [http://www.openheatmap.com/ (Links to an external site.)](http://www.openheatmap.com/)

**Text Analytics Resources**

If you're thinking about a project that involves text (e.g. twitter, facebook, text from web pages, etc) there's an online API service from datumbox that might be useful for doing text analytics: it supports keyword extraction, about 14 flavors of text classifier, and document similarity.

external link: [http://www.datumbox.com/machine-learning-api/ (Links to an external site.)](http://www.datumbox.com/machine-learning-api/)

You send it API calls in the form of HTTP requests, and it returns the results in JSON.

I've got an API key for the class:

d3ce53ca1cead4e08490df097c890967

You can use this with the "API Sandbox" to check out the results from various API calls with your sample text.

**Geographic resources**

Bing Maps API external link: [http://www.microsoft.com/maps/developers/web.aspx (Links to an external site.)](http://www.microsoft.com/maps/developers/web.aspx)

Bing Spatial Data Services external link: [http://msdn.microsoft.com/en-us/library/ff701734.aspx (Links to an external site.)](http://msdn.microsoft.com/en-us/library/ff701734.aspx)

Google Time Zone API external link: [https://developers.google.com/maps/documentation/timezone/ (Links to an external site.)](https://developers.google.com/maps/documentation/timezone/)

World Bank API: This was the source for the data in Homework 1. external link: [http://data.worldbank.org/developers/api-overview (Links to an external site.)](http://data.worldbank.org/developers/api-overview)

**Search and Language resources**

Bing Search API external link: [http://sdrv.ms/17Le3x5 (Links to an external site.)](http://sdrv.ms/17Le3x5)

Microsoft Translator API external link: [http://www.microsofttranslator.com/dev/ (Links to an external site.)](http://www.microsofttranslator.com/dev/)

Microsoft Academic search API external link: [http://academic.research.microsoft.com/about/Microsoft%20Academic%20Search%20API%20User%20Manual.pdf (Links to an external site.)](http://academic.research.microsoft.com/about/Microsoft%20Academic%20Search%20API%20User%20Manual.pdf)

Google Books n-gram corpus external link: [http://books.google.com/ngrams (Links to an external site.)](http://books.google.com/ngrams)

Dataset: external link: [http://aws.amazon.com/datasets/8172056142375670 (Links to an external site.)](http://aws.amazon.com/datasets/8172056142375670)

**Web collections**

Common Crawl: • Currently 6 billion Web documents (81 Tb) • Amazon S3 Public Data Set external link: [http://aws.amazon.com/datasets/41740 (Links to an external site.)](http://aws.amazon.com/datasets/41740)

external link: [https://commoncrawl.atlassian.net/wiki/display/CRWL/About+the+Data+Set (Links to an external site.)](https://commoncrawl.atlassian.net/wiki/display/CRWL/About+the+Data+Set)

Award project using Common Crawl: external link: [http://norvigaward.github.io/entries.html (Links to an external site.)](http://norvigaward.github.io/entries.html) Python example: external link: [http://www.freelancer.com/projects/Python-Data-Processing/Python-script-for-CommonCrawl.html (Links to an external site.)](http://www.freelancer.com/projects/Python-Data-Processing/Python-script-for-CommonCrawl.html)

URL Search: external link: [http://urlsearch.commoncrawl.org/?q=research.microsoft.com (Links to an external site.)](http://urlsearch.commoncrawl.org/?q=research.microsoft.com) Downloadable JSON metadata file with the address and offset of the data for each URL

**Business/commercial data** Yelp external link: [http://www.yelp.com/developers/documentation/v2/search\_api (Links to an external site.)](http://www.yelp.com/developers/documentation/v2/search_api)

IMDB external link: [http://mymovieapi.com/ (Links to an external site.)](http://mymovieapi.com/)

Internet Archive (huge, ever-growing archive of the Web going back to 1990s) external link: [http://archive.org/help/json.php (Links to an external site.)](http://archive.org/help/json.php)

**Social media**

Twitter APIs Documentation external link: [https://dev.twitter.com/docs/api/1.1 (Links to an external site.)](https://dev.twitter.com/docs/api/1.1)

Status stream external link: [https://dev.twitter.com/docs/api/1.1/get/statuses/sample (Links to an external site.)](https://dev.twitter.com/docs/api/1.1/get/statuses/sample) Twitter search external link: [https://dev.twitter.com/docs/api/1.1/get/search/tweets (Links to an external site.)](https://dev.twitter.com/docs/api/1.1/get/search/tweets)

Facebook API external link: [https://developers.facebook.com/docs/reference/api/ (Links to an external site.)](https://developers.facebook.com/docs/reference/api/)

(Analyze your own/friends facebook posts, or create an entity)

Flickr API (Photos) external link: [http://stuvel.eu/flickrapi (Links to an external site.)](http://stuvel.eu/flickrapi) python-flickrapi module

Freebase: a database of the world’s people, places, and other entities and their relationships. external link: [https://developers.google.com/freebase/ (Links to an external site.)](https://developers.google.com/freebase/) (API) external link: [https://developers.google.com/freebase/data (Links to an external site.)](https://developers.google.com/freebase/data) (Data dump)

Freebase annotation of the ClueWeb web collection: external link: [http://googleresearch.blogspot.com/2013/07/11-billion-clues-in-800-million.html (Links to an external site.)](http://googleresearch.blogspot.com/2013/07/11-billion-clues-in-800-million.html)

**Random inspirational examples**

The programming language popularity visualization I showed in the first lecture external link: [http://www.dataists.com/2010/12/ranking-the-popularity-of-programming-langauges/ (Links to an external site.)](http://www.dataists.com/2010/12/ranking-the-popularity-of-programming-langauges/)

HTTP Archive + BigQuery = Web Performance Answers external link: [http://www.igvita.com/2013/06/20/http-archive-bigquery-web-performance-answers/ (Links to an external site.)](http://www.igvita.com/2013/06/20/http-archive-bigquery-web-performance-answers/)

external link: [http://slackprop.wordpress.com/2013/06/03/on-geek-versus-nerd/ (Links to an external site.)](http://slackprop.wordpress.com/2013/06/03/on-geek-versus-nerd/)

Digital History Hacks: external link: [http://digitalhistoryhacks.blogspot.com/2007/01/exploratory-bibliography.html (Links to an external site.)](http://digitalhistoryhacks.blogspot.com/2007/01/exploratory-bibliography.html)

"The first step is to get a complete set of recommendations via the Amazon API. We pass my blog post through a simple scraper to get the ASINs (Amazon Standard ID Numbers) for each book on the list. In a loop, we then submit each ASIN to Amazon and get back an XML file that includes ASINs for recommended books. We create a big list of (ASIN, ASIN) pairs. Each one of these is a recommendation: customers who bought the first book also bought the second one. Since we will want to play with these data later on, we use a high-level Python module called pickle to save them to disk. Python source code for the first step is here.

Now that we've got a big list of recommendations, the second step is to concentrate on the books that come up most frequently. We unpickle our data then create a list of recommended books and filter out any that appear on the original list. We count the number of times each is recommended and sort to create a frequency list with most frequently recommended books at the top of the list. Python source for the second step is here." (see site)

And follow-on work using Amazon and graph visualization here: external link: [http://babbaging.blogspot.com/2007/04/python-amazon-graphs-oh-my.html (Links to an external site.)](http://babbaging.blogspot.com/2007/04/python-amazon-graphs-oh-my.html)